

THIS OPINION WAS NOT WRITTEN FOR PUBLICATION

The opinion in support of the decision being entered today (1) was not written for publication in a law journal and (2) is not binding precedent of the Board.

Paper No. 11

UNITED STATES PATENT AND TRADEMARK OFFICE

---

BEFORE THE BOARD OF PATENT APPEALS  
AND INTERFERENCES

---

Ex parte MARK R. FUNK, STEVEN R. KUNKEL, MIKKO H. LIPASTI,  
BILHA MENDELSON, ROBERT R. ROEDIGER and WILLIAM J. SCHMIDT

---

Appeal No. 1998-1516  
Application 08/420,540

---

ON BRIEF

---

Before KRASS, JERRY SMITH and BLANKENSHIP, Administrative Patent Judges.

KRASS, Administrative Patent Judge.

DECISION ON APPEAL

This is a decision on appeal from the final rejection of claims 1 through 20, all of the

claims pending in the case.

The invention pertains to cache management mechanisms in a computer system. More particularly, the cache management mechanism of the instant invention intelligently places preload instructions into the instruction stream of a computer system such that, through identification of certain instruction stream constructs, the cache memory gets the right information at the right time. These instruction stream constructs, or predictor constructs, are used to predict the presence of two other stream constructs, the first instruction stream construct predicted involving the loading of an address (address generation construct), and the second instruction stream construct predicted involving the use of the generated address to gain access to the associated information (data load construct).

Representative independent claim 1 is reproduced as follows:

1. A computer apparatus, said computer apparatus comprising:

a first central processing unit, said first central processing unit executing an instruction stream, said instruction stream having a first preload instruction inserted therein, said first preload instruction having been inserted by a compiler at a location proximate to a first predictor construct contained in said instruction stream, said first preload instruction containing a first address for first information that will be needed by said first central processing unit.

The examiner relies on the following reference:

Schlansker et al. (Schlansker)	5,404,484	Apr. 4, 1995
		(filed Sep. 16, 1992)

Claims 1 through 5 and 11 through 15 stand rejected under 35 U.S.C. 102(e) as anticipated by Schlansker. Claims 6 through 10 and 16 through 20 stand rejected under 35 U.S.C. 103 as unpatentable over Schlansker.

Reference is made to the brief and answer for the respective positions of appellants and the examiner.

### OPINION

Independent apparatus claim 1 and its counterpart independent method claim 11 broadly recite a computer having a CPU which executes an instruction stream which comprises a preload instruction. The preloaded instruction is inserted by a compiler at a location “proximate to a first predictor construct” contained in the instruction stream. The preload instruction contains an address for information that will be needed by the CPU.

Schlansker clearly discloses a CPU executing an instruction stream which comprises a preload instruction and the preload instruction contains an address for information that will be needed by the CPU. The only issue before us is whether Schlansker discloses the claimed “first predictor construct.”

The examiner's position is that the load instruction in Schlansker is similar to the claimed "predictor construct" because "Schlansker's disclosed compiler inserts the preload instructions at locations proximate to the load instructions in the instruction stream (see abstract)" [Answer-page 4]. For their part, appellants contend that Schlansker's load instructions are not predictor constructs because the load instructions do not predict down stream instructions and they do not predict an address generation construct, making use, instead, of a previously generated address. It is appellants' view that Schlansker's load instructions "do not predict a down stream data load construct, but are themselves data load constructs. As such, a load instruction is not a predictor construct but is the data load construct that is predicted by a predictor construct" [Brief-page 5, emphasis in the original].

Schlansker never explicitly mentions a "predictor construct." Thus, we must determine if the examiner is reasonable in the assertion that Schlansker's load instruction is a "predictor construct," as claimed. To begin the analysis, we must first determine what is meant by a "predictor construct." Appellants provide us with the answer at page 5 of the instant specification:

Through inspection of the instruction stream, the compiler of the present invention detects the existence of certain instruction stream constructs that foretell the information that the processor will need and when the processor will need the information (referred to hereafter as predictor constructs). Typically, predictor constructs explicitly or

implicitly necessitate the presence of two other types of instruction stream constructs. These latter two instruction stream constructs usually perform two functions: 1) the loading or calculation of an address (referred to hereafter as address generation constructs) and 2) use of the generated address to gain access to the information needed by the processor (referred to hereafter as data load constructs).

Thus, according to appellants' own definition, a "predictor construct" is a certain instruction stream construct that foretells the information that the processor will need and when the processor will need that information. Further, the "predictor construct" requires the presence of instruction stream constructs that usually perform two functions, the loading of an address and the use of that address to gain access to the information needed by the processor.

In view of this definition of "predictor construct," it is our view that the examiner makes a prima facie case of anticipation. As pointed out by the examiner [Answer-pages 7-8], Schlansker, at column 3, indicates that the load instruction specifies the address in memory at which data to be retrieved is stored. This would be an "address generation construct" (since the address in memory is specified) and a "data load construct" (since data is retrieved). Thus, Schlansker discloses the presence of an instruction stream construct performing the two functions set forth in appellants' specification as being necessitated by the predictor constructs. Further, the load instruction in Schlansker is used by the compiler to predict what information is needed by the

processor and at which time the information is needed. Accordingly, we find ourselves in agreement with the examiner that Schlansker's load instruction is a "predictor construct," as defined by appellants.

Appellants argue that the load instructions of Schlansker do not predict a down stream data load construct, but are themselves data load constructs. It may be that Schlansker's load instruction does not predict a down stream data construct in the same manner as does appellants' intended "predictor construct" but, as broadly claimed and as broadly defined by appellants, a "predictor construct" appears to be met by Schlansker's load instruction and appellants have presented no evidence to persuade us otherwise.

Accordingly, we will sustain the rejection of claims 1 through 5 and 11 through 15 under 35 U.S.C. 102(e).

With regard to the rejection of claims 6, 9, 10, 16, 19 and 20 under 35 U.S.C. 103, appellants' sole argument is, again, that Schlansker does not teach, disclose or suggest the claimed "predictor construct." Thus, for the reasons supra, we will also sustain the rejection of these claims under 35 U.S.C. 103.

With regard to the rejection of claims 7, 8, 17 and 18 under 35 U.S.C. 103, appellants argue that Schlansker does not suggest the particular predictor constructs recited by these claims. More particularly, the claims call for the first predictor construct being a “call instruction” and the second predictor construct being “one that loads a table of contents pointer.” The processing of these two particular instruction stream constructs is described at pages 13 et seq. in the specification.

The examiner realizes that Schlansker does not disclose the claimed call instruction or TOC pointer predictor constructs. However, the examiner contends [Answer-pages 5-6] that all possible instructions are covered by, and would have been obvious over, Schlansker. We disagree. Without a specific teaching or suggestion by Schlansker to do so, we can only conclude that the examiner’s finding of obviousness with regard to the call and TOC pointer predictor constructs was arrived at through impermissible hindsight gleaned from appellants’ own disclosure. This is not a proper basis for a conclusion of obviousness under 35 U.S.C. 103. Accordingly, we will not sustain the rejection of claims 7, 8, 17 and 18 under 35 U.S.C. 103.

We have sustained the rejection of claims 1 through 5 and 11 through 15 under 35 U.S.C. 102(e) and we have sustained the rejection of claims 6, 9, 10, 16, 19 and 20 under 35 U.S.C. 103.

Appeal No. 1998-1516  
Application 08/420,540

We have not, however, sustained the rejection of claims 7, 8, 17 and 18 under 35 U.S.C. 103.

Accordingly, the examiner's decision is affirmed-in-part.

No time period for taking any subsequent action in connection with this appeal may be extended under 37 CFR § 1.136(a).

**AFFIRMED-IN-PART**

ERROL A. KRASS	)	
Administrative Patent Judge	)	
	)	
	)	
	)	BOARD OF PATENT
JERRY SMITH	)	
Administrative Patent Judge	)	APPEALS AND
	)	
	)	INTERFERENCES
	)	
HOWARD B. BLANKENSHIP	)	
Administrative Patent Judge	)	

Appeal No. 1998-1516  
Application 08/420,540

EAK:pgg  
Steven W. Roth  
IBM corporation Department 917  
3605 Highway 52 N  
Rochester, MN 55901